

(19)日本国特許庁 (J P)

(12) 公 開 特 許 公 報 (A)

(11)特許出願公開番号

特開平11-345127

(43)公開日 平成11年(1999)12月14日

(51)Int.Cl.<sup>5</sup>

G 0 6 F 9/45

識別記号

F I

G 0 6 F 9/44

3 2 2 H

審査請求 有 請求項の数9 O L (全 16 頁)

(21)出願番号 特願平11-83570

(22)出願日 平成11年(1999)3月26日

(31)優先権主張番号 特願平10-88473

(32)優先日 平10(1998)4月1日

(33)優先権主張国 日本 (J P)

(71)出願人 000005821

松下電器産業株式会社

大阪府門真市大字門真1006番地

(72)発明者 中島 雅逸

大阪府門真市大字門真1006番地 松下電器

産業株式会社内

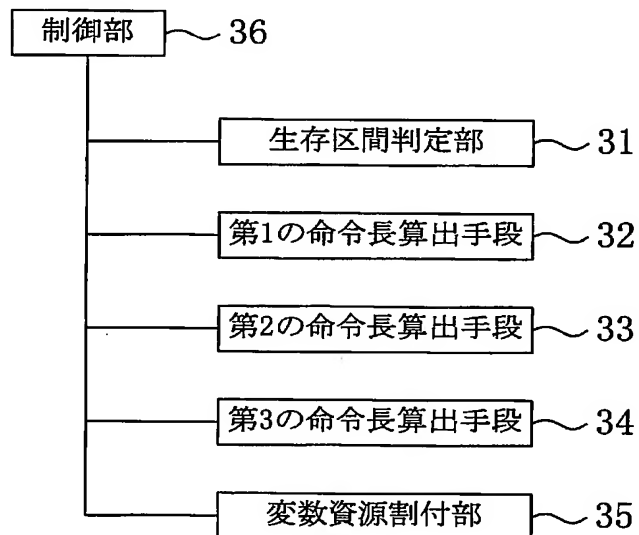
(74)代理人 弁理士 前田 弘 (外1名)

(54)【発明の名称】 コンパイラ

(57)【要約】 (修正有)

【課題】 命令コードサイズの小さい機械語プログラムを生成できるコンパイラを提供する。

【解決手段】 第1の命令列長算出手段32は、ソースプログラム中の各変数を、第1の命令フォーマットで記述された命令を使用して第1のレジスタ資源に割り付けた場合の命令列長を算出する。第2の命令列長算出手段33は、ソースプログラム中の各変数を、前記第1の命令フォーマットとは命令長が異なる第2の命令フォーマットで記述された命令を使用して、前記第1のレジスタ資源とは異なる第2のレジスタ資源に割り付けた場合の命令列長を算出する。前記算出結果に応じて資源割り付けを行う。



## 【特許請求の範囲】

【請求項 1】 複数の命令からなるソースプログラムを入力し、このソースプログラムを機械語プログラムに翻訳するコンパイラであって、

前記ソースプログラム中の各変数を、第 1 の命令フォーマットで記述された命令を使用して第 1 のレジスタ資源に割り付けた場合の命令列の長さを算出する第 1 の命令列長算出手段と、

前記ソースプログラム中の各変数を、前記第 1 の命令フォーマットとは命令長が異なる第 2 の命令フォーマットで記述された命令を使用して、前記第 1 のレジスタ資源とは異なる第 2 のレジスタ資源に割り付けた場合の命令列の長さを算出する第 2 の命令列長算出手段とを備え、前記第 1 及び第 2 の命令長算出手段が算出した命令長算出結果を用いて資源割り付けを行うことを特徴とするコンパイラ。

【請求項 2】 前記ソースプログラム中の各変数を、メモリに割り付けた場合の命令列の長さを算出する第 3 の命令列長算出手段を備え、

前記第 1、第 2 及び第 3 の命令長算出手段が算出した命令長算出結果を用いて資源割り付けを行うことを特徴とする請求項 1 記載のコンパイラ。

【請求項 3】 前記第 1 及び第 2 の命令長算出手段が算出した命令長算出結果を比較して、命令列長が小さい資源割り付けを行うことを特徴とする請求項 1 記載のコンパイラ。

【請求項 4】 前記第 1 のレジスタ資源は、前記第 2 のレジスタ資源の一部であることを特徴とする請求項 1 記載のコンパイラ。

【請求項 5】 前記第 1 の命令フォーマットの命令長は、前記第 2 のフォーマット命令フォーマットの命令長より短いことを特徴とする請求項 4 記載のコンパイラ。

【請求項 6】 前記ソースプログラム中の各変数の中で、使用頻度が高い変数を優先的に第 1 の命令フォーマットで記述された命令を使用して第 1 のレジスタ資源に割り付けることを特徴とする請求項 1 記載のコンパイラ。

【請求項 7】 前記ソースプログラム中の各変数について、前記第 1 のレジスタ資源に割り付けられた変数进行操作する場合には、前記第 1 の命令フォーマットを優先的に使用してその操作を記述することを特徴とする請求項 6 記載のコンパイラ。

【請求項 8】 前記プログラム中の各変数の中で、使用頻度が高い変数、及びこの変数と共に使用される変数を、優先的に第 1 のレジスタ資源に割り付けることを特徴とする請求項 6 記載のコンパイラ。

【請求項 9】 複数の命令からなるソースプログラムを翻訳した機械語プログラムを記録したコンピュータ読み取り可能な記録媒体であって、前記機械語プログラムは、

第 1 のレジスタ資源を用いて第 1 の命令フォーマットで記述された命令と、

前記第 1 のレジスタ資源とは異なる第 2 のレジスタ資源を用いて、前記第 1 の命令フォーマットとは命令長が異なる第 2 の命令フォーマットで記述された命令とが混在し、前記第 1 の命令フォーマット及び第 2 の命令フォーマットは前記命令中の特定の領域の値により識別されることを特徴とする記録媒体。

## 【発明の詳細な説明】

## 【0001】

【発明の属する技術分野】本発明は、高級プログラミング言語で書かれたソースプログラムを機械語プログラムに翻訳するコンパイラに関する。

## 【0002】

【従来の技術】近年、C 言語などの高級プログラミング言語でプログラムを記述し、プログラムの開発効率を高めることが盛んに行われている。高級プログラミング言語を用いることによって、プログラマは、プログラム中の数値の保持、演算、転送等の処理を、変数を用いた演算（ステップ）によって、任意に定義し、また、必要な個数だけ用いることができるため、プログラマは自由にプログラムを記述することができる。また記述されたプログラム（以下、ソースプログラムと称する）は、コンパイラによってコンパイルされることにより、コンピュータが理解可能な機械語プログラムとなる。この機械語プログラム中の演算は、機械語命令によって表現され、また、当該機械語命令は、レジスタ又はメモリをオペランドとするので、前記変数には、レジスタ又はメモリを割り付ける必要がある。この割り付け処理は、資源割り付け処理と称される。この資源割り付け処理が最適であれば、前記機械語プログラムのコードサイズを最小となる。

【0003】一般的には、各変数に対して、レジスタを割り付けた場合の方が、メモリを割り付けた場合に比較して、コードサイズ的にも、実行時間的にも良好な結果が得られるが、レジスタは使用できる個数が少数であるため、如何に効率良くレジスタ資源の割り付けを行って、レジスタをオペランドとする機械語命令を使用するかということが、資源割り付け処理における最適化である。従来の資源割り付けの最適化方式としては、各々の変数が格納されている値が有効となる区間（変数の生存区間と称する。）に基づき、同一レジスタに割り付け変数はどれとどれであるかを検査し、その結果を用いて資源割り付けを行う方式が採用されている。

【0004】一方、本願発明者は、特願平 1 0 - 5 9 6 8 0 において、以下のような 2 種の命令フォーマット及びレジスタモデルを使用するデータ処理装置を提案している。図 1 0 から図 2 0 において、第 1 の命令フォーマットの概要を示す。

【0005】第 1 の命令フォーマットは、最小命令語長

を1バイトとした可変長命令であり、レジスタアドレス指定フィールドとしては、2ビットのフィールドが使用される。従って、1つのレジスタアドレス指定フィールドで4個のレジスタが指定可能である。本アーキテクチャでは、4個のアドレスレジスタと、4個のデータレジスタが定義される。命令動作として、アドレスレジスタを使用するか、データレジスタを使用するかを区別することにより、命令で前記合計8個のレジスタを使用できるように構成される。

【0006】図10は、最小命令語長である1バイト目の第1の命令フィールドが、オペレーション指定フィールドと、任意の数のレジスタアドレス指定フィールドとから構成される第1の命令フォーマット(1)のビット割り付けを示す。

【0007】第1の命令フォーマット(1)-(a)は、第1の命令フィールド内に2ビットのレジスタアドレス指定フィールドを2フィールド含み、最小命令語長である1バイトで構成される命令フォーマットであり、2つのオペランドが指定可能な命令フォーマットである。

【0008】第1の命令フォーマット(1)-(b)は、第1の命令フィールド内に2ビットのレジスタアドレス指定フィールドを2フィールド含み、更に付加情報フィールドを追加して、合計2バイト以上の命令語長を持つ命令フォーマットである。

【0009】第1の命令フォーマット(1)-(c)は、第1の命令フィールド内に2ビットのレジスタアドレス指定フィールドを1つ含み、最小命令語長である1バイトで構成される命令フォーマットであり、1つのオペランドが指定可能な命令フォーマットである。

【0010】第1の命令フォーマット(1)-(d)は、第1の命令フィールド内に2ビットのレジスタアドレス指定フィールドを1フィールド含み、更に付加情報フィールドを追加した2バイト以上の命令語長を持つ命令フォーマットである。

【0011】第1の命令フォーマット(1)-(e)は、第1の命令フィールド内にレジスタアドレス指定フィールドを含まず、最小命令語長である1バイトで構成される命令フォーマットであり、アドレスを用いたオペランド指定不可能な命令フォーマットである。

【0012】第1の命令フォーマット(1)-(f)は、第1の命令フィールド内にレジスタアドレス指定フィールドを含まず、更に付加情報フィールドを追加した2バイト以上の命令語長を持つ命令フォーマットである。

【0013】図11は、図10で示された個々のビット割り付けについて、具体的な命令のリストの一部を示したものである。左側に命令のニーモニックを、右側に命令の動作を各々示している。

【0014】図12は、最小命令語長である1バイト目の第1の命令フィールドが命令語長指定フィールドからなり、第2の命令フィールドが、オペレーション指定フ

ィールドと、任意の数のレジスタアドレス指定フィールドとから構成される第1の命令フォーマット(2)のビット割り付けを示したものである。

【0015】第1の命令フォーマット(2)-(a)は、第2の命令フィールド内に2ビットのレジスタアドレス指定フィールドを2フィールド含み、2バイトで構成される命令フォーマットであり、2つのオペランドが指定可能な命令フォーマットである。

【0016】第1の命令フォーマット(2)-(b)は、第2の命令フィールド内に2ビットのレジスタアドレス指定フィールドを2フィールド含み、更に付加情報フィールドを追加して、合計3バイト以上の命令語長を持つ命令フォーマットである。

【0017】第1の命令フォーマット(2)-(c)は、第2の命令フィールド内に2ビットのレジスタアドレス指定フィールドを1つ含み、2バイトで構成される命令フォーマットであり、1つのオペランドが指定可能な命令フォーマットである。

【0018】第1の命令フォーマット(2)-(d)は、第2の命令フィールド内に2ビットのレジスタアドレス指定フィールドを1フィールド含み、更に付加情報フィールドを追加した3バイト以上の命令語長を持つ命令フォーマットである。

【0019】第1の命令フォーマット(2)-(e)は、第2の命令フィールド内にレジスタアドレス指定フィールドを含まず、2バイトで構成される命令フォーマットであり、アドレスを用いたオペランド指定が不可能な命令フォーマットである。

【0020】第1の命令フォーマット(2)-(f)は、第2の命令フィールド内にレジスタアドレス指定フィールドを含まず、更に、付加情報フィールドを追加した2バイト以上の命令語長を持つ命令フォーマットである。

【0021】図13は、図12で示された個々のビット割り付けについて、具体的な命令のリストの一部を示したものである。左側に命令のニーモニックを、右側に命令の動作を各々示している。

【0022】従って、前記図10から図13に示した第1の命令フォーマットは、第1の命令フィールドを基本命令語長とし、第1から第Mの命令フィールドを最長命令語長Mとして、N命令語長(Nは1～M間での整数)の可変長命令を特定するものであって、最小命令語長が1バイトであるという他にない特徴を備え、プログラムサイズの縮小に適した命令フォーマットとなっている。

【0023】図14は、本データ処理装置に備える第1のレジスタファイル220を示す。この第1のレジスタファイル220は、4個のアドレスレジスタA0～A3と、4個のデータレジスタD0～D3と、スタックポインタSP223と、内部のステータス情報及び制御情報を保持するPSW(Processor Status Word)224と、プログラムカウンタPC225とを含む。

【0024】更に、図15は、前記第1のレジスタファイル220のアドレスレジスタA0～A3及びデータレジスタD0～D3へのアクセスについて、より詳細に示した図である。同図は、命令の中で指定されるレジスタ名と、レジスタアドレス指定フィールドで指定される命令コードでのビット割り付けと、物理的なレジスタにアクセスするための物理的なレジスタ番号、及びアクセスする対象となる物理的なレジスタ名を一覧にして示したものである。

【0025】図15に示すように、第1の命令フォーマットにおいては、4個のアドレスレジスタA0～A3へアクセスするために命令内に指定される命令アドレス指定フィールドと、4個のデータレジスタD0～D3へアクセスするために命令内に指定される命令アドレス指定フィールドとは、全く同一である。即ち、レジスタのアドレスを指定するために2ビット命令アドレス指定フィールドを使用し、命令動作そのものでアドレスレジスタにアクセスするか、データレジスタにアクセスするかを区別させている。

【0026】次に、本アーキテクチャの基本命令フォーマットである前記図10及び図12の第1の命令フォーマットに対して、追加拡張する第2の命令フォーマットのビット割り付けを図16に示す。

【0027】図16に示した第2の命令フォーマットのビット割り付けは、最小命令語長である1バイト目の第1の命令フィールドが命令語長指定フィールドからなり、第2及び第3の命令フィールドが、オペレーション指定フィールドと、任意の数のレジスタアドレス指定フィールドとから構成される。第2の命令フォーマットにおけるレジスタアドレス指定フィールドは4ビットから

構成される。

【0028】図16において、第2の命令フォーマット(a)は、第3の命令フィールド内に4ビットのレジスタアドレス指定フィールドを2フィールド含み、3バイトで構成される命令フォーマットであり、2つのオペランドが指定可能な命令フォーマットである。

【0029】第2の命令フォーマット(b)は、第3の命令フィールド内に4ビットのレジスタアドレス指定フィールドを2フィールド含み、更に、付加情報フィールドを追加して、合計4バイト以上の命令語長を持つ命令フォーマットである。

【0030】第2の命令フォーマット(c)は、第3の命令フィールド内に4ビットのレジスタアドレス指定フィールドを1つ含み、3バイトで構成される命令フォーマットであり、1つのオペランドが指定可能な命令フォーマットである。

【0031】第2の命令フォーマット(d)は、第3の命令フィールド内に4ビットのレジスタアドレス指定フィールドを1フィールド含み、更に、付加情報フィールドを追加した4バイト以上の命令語長を持つ命令フォー

マットである。

【0032】従って、前記第2の命令フォーマットも、第1の命令フィールドを基本命令語長とし、第1から第Mの命令フィールドを最長命令語長Mとして、N命令語長(Nは1～M間での整数)の可変長命令を特定する。

【0033】図17は、図16で示された個々のビット割り付けについて、具体的な命令のリストの一部を示したものである。左側に命令のニーモニックを、右側に命令の動作を各々示している。ニーモニックの中で、Rm、Rn、又はRiは、レジスタアドレスの指定を表すが、指定できるレジスタとして、図18に示されるような第2のレジスタファイル120が定義され、4個のアドレスレジスタA0～A3、4個のデータレジスタD0～D3、及び8個の拡張レジスタE0～E7から構成される16個の汎用レジスタである。更に、この第2のレジスタファイル120は、スタックポインタSP122と、内部のステータス情報及び制御情報を保持するPSW(Processor Status Word)123と、プログラムカウンタPC124とを含んでいる。

【0034】図19は、第1の命令フォーマットで定義された命令を実行する際に、命令の中で指定されるレジスタ名と、レジスタアドレス指定フィールドで指定される命令コード上でのビット割り付けと、物理的なレジスタにアクセスするための物理的なレジスタ番号、及び、アクセスする対象となる物理的なレジスタ名を一覧にして示したものである。第1の命令フォーマットでは、レジスタ指定フィールドは2ビットしかないが、汎用レジスタは16個で4ビットのアドレスでアクセスする必要がある関係から、アドレスの変換をする必要がある。例えば、アドレスレジスタA0をアクセスする際には、物理的なアドレス番号として“1000”が、データレジスタD1をアクセスする際には、物理的なアドレス番号として、“1101”を生成し、汎用レジスタファイル121に出力する必要がある。

【0035】図20は、第2の命令フォーマットで定義された命令を実行する際に、命令の中で指定されるレジスタ名と、レジスタアドレス指定フィールドで指定される命令コード上でのビット割り付けと、物理的なレジスタにアクセスするための物理的なレジスタ番号、及び、アクセスする対象となる物理的なレジスタ名を一覧にして示したものである。第2の命令フォーマットでは、4ビットのレジスタアドレス指定フィールドを有しているので、その4ビットをそのまま物理的なレジスタ番号として指定することになる。

【0036】

【発明が解決しようとする課題】本願発明者が特願平10-59680において提案したようなデータ処理装置においては、従来のコンパイラが行っているように単にレジスタをメモリよりも優先して資源割り付けを行うと、以下のような問題が存在する。

【0037】1) 第1のレジスタファイル(レジスタ資源)に割り付けられた変数を第1の命令フォーマットで記述された命令を使用して処理する場合と、第2のレジスタファイルに割り付けられた変数を第2の命令フォーマットで記述された命令を使用して処理する場合とで命令長が異なるため、両者の何れをも優先的に扱わず均一に扱うことでは、命令コードサイズは最小にならない。即ち、従来のコンパイラでは、各変数を第1のレジスタファイルに優先的に割り付けるか、又は第2のレジスタファイルに優先的に割り付けるかについて考慮していない。例えば、各変数をその出現順に第2のレジスタファイルに割り付け、これ等変数を第2の命令フォーマットで記述された命令を使用して処理する場合には、第2の命令フォーマットの命令長が第1の命令フォーマットの命令長よりも長い以上、命令コードサイズは大きくなる。

【0038】2) 第2の命令フォーマットで記述された命令を使用して変数を処理する場合に対して、データをメモリからレジスタに転送するデータ転送命令を含む命令列では、第1の命令フォーマットを使用して変数を処理した方が、命令数は増加しても命令長は短くなる場合がある。従って、単純に、変数をメモリよりもレジスタ資源に優先的に割り付けることでは、コードサイズは最小にならない。

【0039】本発明の目的は、命令フォーマットによって扱えるレジスタ資源が異なり、且つ使用する命令フォーマットに応じて命令長が異なる命令セットを有するプロセッサにおいて、命令コードサイズが小さい機械語プログラムを生成するコンパイラを提供することにある。

#### 【0040】

【課題を解決するための手段】前記課題を解決するために、本発明では、頻繁に使用される変数が、命令長の短いフォーマットで使用可能なレジスタに割り付けられ、且つそれ等変数が前記命令長の短いフォーマットを使用して処理される機械語プログラムを得るようにする。

【0041】すなわち、請求項1記載の発明のコンパイラは、複数の命令からなるソースプログラムを入力し、このソースプログラムを機械語プログラムに翻訳するコンパイラであって、前記ソースプログラム中の各変数を、第1の命令フォーマットで記述された命令を使用して第1のレジスタ資源に割り付けた場合の命令列の長さを算出する第1の命令列長算出手段と、前記ソースプログラム中の各変数を、前記第1の命令フォーマットとは命令長が異なる第2の命令フォーマットで記述された命令を使用して、前記第1のレジスタ資源とは異なる第2のレジスタ資源に割り付けた場合の命令列の長さを算出する第2の命令列長算出手段とを備え、前記第1及び第2の命令列長算出手段が算出した命令列長算出結果を用いて資源割り付けを行うことを特徴とする。

【0042】請求項2記載の発明は、前記請求項1記載

のコンパイラにおいて、前記ソースプログラム中の各変数を、メモリに割り付けた場合の命令列の長さを算出する第3の命令列長算出手段を備え、前記第1、第2及び第3の命令列長算出手段が算出した命令列長算出結果を用いて資源割り付けを行うことを特徴とする。

【0043】請求項3記載の発明は、前記請求項1記載のコンパイラにおいて、前記第1及び第2の命令列長算出手段が算出した命令列長算出結果を比較して、命令列長が小さい資源割り付けを行うことを特徴とする。

10 【0044】請求項4記載の発明は、前記請求項1記載のコンパイラにおいて、前記第1のレジスタ資源は、前記第2のレジスタ資源の一部であることを特徴とする。

【0045】請求項5記載の発明は、前記請求項4記載のコンパイラにおいて、前記第1の命令フォーマットの命令長は、前記第2のフォーマット命令フォーマットの命令長より短いことを特徴とする。

20 【0046】請求項6記載の発明は、前記請求項1記載のコンパイラにおいて、前記ソースプログラム中の各変数の中で、使用頻度が高い変数を優先的に第1の命令フォーマットで記述された命令を使用して第1のレジスタ資源に割り付けることを特徴とする。

【0047】請求項7記載の発明は、前記請求項6記載のコンパイラにおいて、前記ソースプログラム中の各変数について、前記第1のレジスタ資源に割り付けられた変数を操作する場合には、前記第1の命令フォーマットを優先的に使用してその操作を記述することを特徴とする。

30 【0048】請求項8記載の発明は、前記請求項6記載のコンパイラにおいて、前記プログラム中の各変数の中で、使用頻度が高い変数、及びこの変数と共に使用される変数を、優先的に第1のレジスタ資源に割り付けることを特徴とする。

【0049】請求項9記載の発明の記録媒体は、複数の命令からなるソースプログラムを翻訳した機械語プログラムを記録したコンピュータ読み取り可能な記録媒体であって、前記機械語プログラムは、第1のレジスタ資源を用いて第1の命令フォーマットで記述された命令と、前記第1のレジスタ資源とは異なる第2のレジスタ資源を用いて、前記第1の命令フォーマットとは命令長が異なる第2の命令フォーマットで記述された命令とが混在し、前記第1の命令フォーマット及び第2の命令フォーマットは前記命令中の特定の領域の値により識別されることを特徴とする。

40 【0050】以上の構成から、本発明では、命令フォーマットの種類に応じて扱えるレジスタ資源が異なり、且つ命令フォーマットの種類に応じて命令長が異なる命令セットを有するプロセッサにおいて、各変数を第1の命令フォーマットで記述された命令を使用して第1のレジスタ資源に割り付けた場合の命令列長の長さ、各変数を第2の命令フォーマットで記述された命令を使用して

第2のレジスタ資源に割り付けた場合の命令列長の長さ  
とが算出され、その算出結果に応じて適切な資源割り  
付けが行われるので、コードサイズの小さい機械語プロ  
グラムが生成される。

#### 【0051】

【発明の実施の形態】以下、本発明の実施の形態につ  
いて、図1から図5を用いて説明する。

【0052】最初に本発明におけるコンパイラが対象と  
するプロセッサの機械語命令セットについてであるが、  
命令のフォーマットによって扱えるレジスタ資源が異な  
り、且つ、命令のフォーマットに応じて命令長が異なる  
ような機械語命令セットを有するプロセッサとして、特  
願平10-59680において提案した2種の命令フォ  
ーマットを有するデータ処理装置とする。

【0053】図1は、本発明の実施の形態におけるコン  
パイラの構成図である。コンパイラは、構文解析手段1  
と、最適化手段2と、資源割り付け手段3と、コード生  
成手段4とで構成されている。

【0054】構文解析手段1は、ファイルとして記憶さ  
れているソースプログラム5の字句解析、構文解析、及  
び意味解析を行う。解析結果は、中間言語プログラムと  
して出力される。

【0055】最適化手段2は、最終的に生成される機械  
語プログラム6のプログラムサイズを削減し、実行時間  
を短縮させる目的で中間言語プログラムの最適化を行  
う。

【0056】資源割り付け手段3は、プログラムの変数  
の生存区間を求め、生存区間毎に各変数に対して資源で  
あるレジスタやメモリを割り付けると共に、同一動作に  
対しての最適な命令の割り付けも行う。

【0057】コード生成手段4は、最適化された中間言  
語プログラムを、資源割り付け手段3の割り付け結果に  
従って、ターゲットマシンの機械語命令に変換し、機械  
語プログラム6として出力する。

【0058】尚、構文解析手段1、最適化手段2、コー  
ド生成手段4については公知であるので、詳細な説明を  
省略する。

【0059】図2は、本発明の実施の形態におけるコン  
パイラの資源割り付け手段3の構成図である。同図にお  
いて、31は各変数に対しての生存区間を判定する生存  
区間判定部である。32は変数を第1の命令フォーマ  
ットを持つ命令群を用いてレジスタに割り付けた場合の命  
令列長を算出する第1の命令長算出手段である。第1の  
命令フォーマットでは、一部のレジスタしか扱うことが  
できない。33は変数を第2の命令フォーマットを持つ  
命令群を用いてレジスタに割り付けた場合の命令長を算  
出する第2の命令長算出手段である。第2の命令フォー  
ーマットでは全てのレジスタ資源を扱うことができる。3  
4は変数をメモリに割り付けた場合の命令長を算出する  
第3の命令長算出手段である。35は生存区間判定部3

1の生存区間の判定結果と、第1から第3の命令長算出  
手段32～34に応じて、変数資源の割り付けを行う変  
数資源割り付け部である。36は資源割り付け手段3の  
制御部36を除く全体を制御する制御部である。

【0060】以上のように構成されたコンパイラの資源  
割り付けについて、以下、図面を用いて、その動作を述  
べる。図3は、本発明の実施の形態におけるコンパイラ  
の制御部36の資源割り付けの処理フローを示したフロ  
ーチャートである。

10 【0061】ここで、コンパイラの資源割付けを行う対  
象とする中間コードプログラム例を図4に示す。図4に  
示された中間コードs1～s8を今回の資源割付けの対  
象となる基本ブロックとする。説明を簡単にするため  
に、ここでは、基本ブロック内の資源割り付けについて  
説明するが、基本ブロックを越えての資源割付けにおい  
ても同様に説明できる。

【0062】また、説明を簡単にするために、レジスタ  
資源の割り付けに関しては、以下の条件で行うものとす  
る。

20 【0063】1) 第1の命令フォーマットで操作できる  
レジスタ資源(A0～A3, D0～D3)の中で、アド  
レスレジスタA0～A3については、ポインタとして使  
用するものとし、データ変数を格納しないものとする。

【0064】2) 第1の命令フォーマットで操作できる  
レジスタ資源(A0～A3, D0～D3)の中で、デー  
タレジスタD0～D3は、全てデータ変数の格納及び操  
作に使用できるが、そのうち、レジスタD0、D1につ  
いてはワークレジスタとして使用し、変数を割り付けな  
いものとする。

30 【0065】3) 第2の命令フォーマットで操作できる  
レジスタ資源(A0～A3, D0～D3, E0～E7)  
の中で、拡張レジスタE0～E7については、データで  
もアドレスでも自由に使用できるものとする。

【0066】これ等の条件は、説明を簡単にするために  
定めたものであって、本発明に何らの制約を与えるもの  
ではない。

【0067】このフローチャートに従って、図4におけ  
る中間コードプログラムについて資源割り付けを行う場  
合の処理について説明する。

40 【0068】まず、ステップ301では、基本ブロック  
内部の全ての変数について、個々の変数の生存区間や参  
照頻度を調べる。ここで、レジスタ割付けの対象となる  
のは、第1の命令フォーマットで操作できるレジスタ資  
源のみ、具体的には、レジスタA0～A3、D0～D3  
のみである。その結果によって、変数資源割付け部35が  
第1の命令フォーマットで操作できるレジスタ資源に対  
して、レジスタ割付け可能かどうかを検査する。ここ  
では、どのような手法を用いてこれ等のレジスタに変数  
を割り付けるかということが重要ではなく、同一のオペ  
レーションに対して第1の命令フォーマットの命令長の方

が第2の命令フォーマットの命令長に比べて小さいので、第1の命令フォーマットで扱えるレジスタ資源のみを対象にしたレジスタ資源割り付けを行うということが最も重要である。

【0069】次に、ステップ302では、ステップ301の検査結果をもとに、第1の命令フォーマットで操作できるレジスタ資源のみで全ての変数が資源割り付け可能かどうかを判断する。

【0070】可能な場合は、第1の命令フォーマットで操作できるレジスタ資源のみに変数を割り付けて、資源割り付けを終了する(ステップ303)。

【0071】通常の基本ブロック内には、多くの変数が存在するので、ほとんどの場合は、第1の命令フォーマットで操作できるレジスタ資源に割り付けられない変数が存在する。ステップ302では、第1の命令フォーマットで操作できるレジスタ資源で全ての変数が割り付けられない場合、先ず、第1の命令フォーマットで操作できるレジスタ資源を優先的に割り付けを行う。この優先的な割り付け処理を以下に説明する。

【0072】図5は、ステップ301で、第1の命令フォーマットで操作できるレジスタ資源に割り付ける変数を決定するために調べた、基本ブロック内の変数の生存区間と、参照頻度とを示した図である。この結果、最も参照頻度の高い変数であるp1とp2を第1の命令フォーマットで操作できるレジスタの内、レジスタD2とD3に割り付けるものとする。ここで、どの変数をレジスタに割り付けるかどうかの決定は、扱えるレジスタ資源の数と、各変数の生存区間、参照頻度によって決定されたものである。この場合、既述した条件によりレジスタD0及びD1は使用できないので、使用できるレジスタ資源はレジスタD2とD3との2つであり、最も使用頻度の高い2つの変数を割り付けている。

【0073】次に、ステップ304では、変数資源割り付け部35が第1の命令フォーマットで操作できるレジスタ資源に割り付けできなかった残りの変数について、第1の命令フォーマットでは操作できず、第2の命令フォーマットによってのみ操作できるレジスタ資源、具体的には、拡張レジスタE0～E7に各変数を割り付けるか、メモリ資源に割り付けるかの第1の組み合わせ(資源の割り付け)について場合分けを行う。

【0074】更に、ステップ304では、制御部36が第1の組み合わせの中でレジスタ資源に割り付けた場合には、ワークレジスタD0、D1を利用して、第1の命令フォーマットを使用するのが適当か、又はワークレジスタD0、D1を利用せずに第2の命令フォーマットを使用するのが適当かも含めて、第2の組み合わせ(命令の割り付け)を考える。拡張レジスタE0～E7及びレジスタD2、D3を使用した特定の第1の組み合わせに対する第2の組み合わせの例を図7～図9に示す。これ等の図面の説明は後に詳述する。最終的には、全ての組

み合わせについて、命令列長を算出するので、ここで、どの組み合わせを選択するかは問題にならない。

【0075】以下に説明するステップ305～ステップ307では、前記第2の組み合わせの各々(図7～図9)に対して命令列長の算出を行う。

【0076】つまり、ステップ305では、第1の命令フォーマットを使用してレジスタに資源割り付けした命令列長を算出する。ステップ306では、第2の命令フォーマットを使用してレジスタに資源割り付けした命令列長を算出する。

【0077】同様に、ステップ307では、レジスタ資源ではなく、メモリ資源に変数を割り付け、第1の命令フォーマットを割り付けた場合の命令列長を算出する。ここでは、説明を簡単にするために、メモリ資源に変数を割り付けた場合のアドレスは、16ビットの絶対アドレスで指定できるものとし、その変数に対してのメモリ/レジスタ間移動命令は、3バイトとする。またステップ307では、レジスタ資源ではなく、メモリ資源に変数を割り付け、第2の命令フォーマットを割り付けた場合の命令列長を算出する。ここでは、説明を簡単にするために、メモリ資源に変数を割り付けた場合のアドレスは、16ビットの絶対アドレスで指定できるものとし、その変数に対してのメモリ/レジスタ間移動命令は、4バイトとする。

【0078】ステップ308では、前記ステップ305～ステップ307の結果を用いて、この組み合わせにおける基本ブロック全体の命令列長を算出する。

【0079】前記処理を全ての組み合わせにおいて実行し(ステップ309)、ステップ310では、その結果、基本ブロック内の命令列長が最小となる組み合わせを選択する)。

【0080】以上のような処理で機械語プログラムサイズを最小化するコンパイラの処理について、図を用いて更に詳細に説明する。

【0081】先ず、図6に、各変数のレジスタ割り付けについて第1の命令フォーマットによってアクセス可能なレジスタ資源と、それ以外のレジスタ資源について、優先度を付けずに出現順にレジスタ資源への割り付けを行った場合(従来の方法)の変数の資源割り付けと、その割り付けに対応して命令を割り付けた場合の機械語命令プログラムを示す。この場合の基本ブロックに対する命令列長は46バイトになる。

【0082】次に、図7に、図5で示された変数の生存区間と出現頻度をもとに、図3のステップ302の処理により、第1の命令フォーマットで操作可能なレジスタのみ(この場合、具体的には、D2レジスタとD3レジスタ)に、優先的にレジスタ割り付けを行い、その結果、変数p1をレジスタD2に、変数p2をレジスタD3に割り付けた場合の変数の資源割り付けと、その割り付けに対応して命令を割り付けた場合の機械語命令プログラ



ムを示す。この場合の基本ブロックに対する命令列長は42バイトとなり、図6で示した場合に比べて、4バイトのコードサイズの削減が図れていることが判る。

【0083】更に、図8に、図7で説明した場合とレジスタ資源の割り付け方は同様であるが、命令フォーマットとして第2の命令フォーマットを選択する代りに第1の命令フォーマットを選択して、命令の割り付け方を変更した場合の機械語命令プログラムを示す。この場合、基本命令ブロックに対するトータルの命令長自体は、図7の場合より2バイト増加して、44バイトになっているが、個々の中間言語毎に機械語命令列長を比較していくと、中間言語s1及び中間言語s8に関しては、これ等言語s1、s8が第1の命令フォーマットで記述された命令を含んで、各々、1バイトずつ機械語命令長が減少していることが判る。ここで、中間言語s1では、使用頻度の高い変数p1及びp2が第1のレジスタファイル220内のレジスタD2、D3に各々割り付けられる。また、これ等変数p1、p2と共に使用される変数t1も第1のレジスタファイル220内のワークレジスタD0に一時的に割り付けられる。これにより、命令mov D2,D0及び命令add D3,D0は、各々、第1の命令フォーマットを使用して1バイトで記述される。

【0084】最後に、図7及び図8で示した結果をもとに、レジスタ資源の割り付け、及び命令の割り付けを最適化した機械語命令プログラムを図9に示す。図9では、図7及び図8の各中間言語の中で機械語命令長が短い方の中間言語、即ち、中間言語s2～s7では図7の中間言語s2～s7が、中間言語s1、s8では図8の中間言語s1、s8が採用される。尚、図7の中間言語s2、s3と図8の中間言語s2、s3とは機械語命令長が同一であるが、図7の中間言語s2、s3、即ち命令数が少なく高速処理の可能な方の中間言語が採用される。その結果、図9では、図7の状態から、更に2バイトのコード削減が実現されており、図3及び図4で示した中間言語プログラムに対して、最もコードサイズの小さい機械語命令プログラムが生成されている。

【0085】前記図9の機械語命令プログラムは、中間言語s1の命令mov D2,D0、命令add D3,D0、及び中間言語s8の命令add 5,D0が第1のレジスタファイル220内のレジスタD0、D2、D3を用いて第1の命令フォーマットで記述され、他の中間言語s2～s7が第2のレジスタファイル120を用いて第2の命令フォーマットで記述されていて、この機械語命令プログラムは、コンピュータ読み取り可能な記録媒体に記録される。

【0086】尚、以上の具体例では、基本ブロックの各変数p1、p2、t1～t8をレジスタD2、D3、E0～E7に割り付けたが、資源割り付けが必要な変数が割り付け可能なレジスタの本数を越えて同時に存在する場合には、これ等変数は前記各レジスタとメモリ資源とに割り付けられる。この場合は、図3のステップS30

7においてメモリ資源に割り付けた場合の命令列長が算出される。このように変数の一部をメモリ資源に割り付ける場合、データ転送命令を含む命令列では、第1のレジスタファイル220内の何れかのレジスタを一時的に使用して第1の命令フォーマットが使用される。これにより、命令数は増加するが、命令列長は短くなり、命令コードサイズは小さくなる場合がある。

【0087】

【発明の効果】以上説明したように、本発明のコンパイラによれば、命令フォーマットの種類に応じて扱えるレジスタ資源が異なり、且つ命令フォーマットの種類に応じて命令長が異なる命令セットを有するプロセッサにおいて、コードサイズの小さい機械語プログラムを生成できるコンパイラを提供できるという顕著な効果が得られる。

【図面の簡単な説明】

【図1】本発明の一実施の形態によるコンパイラの構成を示す構成図である。

【図2】同実施形態によるコンパイラの資源割り付け手段の構成を示す構成図である。

【図3】同実施形態によるコンパイラの資源割り付け手段における資源割り付けに関する処理フローを説明するフローチャート図である。

【図4】中間言語プログラムの一例を示す図である。

【図5】中間言語プログラム中の各変数の生存区間と参照頻度を示した図である。

【図6】変数の資源割付けとそれに対応した機械語命令プログラムの一例を示す図である。

【図7】変数の資源割付けとそれに対応した機械語命令プログラムの一例を示す図である。

【図8】変数の資源割付けとそれに対応した機械語命令プログラムの一例を示す図である。

【図9】変数の資源割付けとそれに対応した機械語命令プログラムの一例を示す図である。

【図10】本発明によるコンパイラが対象とするデータ処理装置の第1の命令フォーマット(1)を示す図である。

【図11】同データ処理装置の第1の命令フォーマット(1)の命令のリストの一部を示す図である。

【図12】同データ処理装置の第1の命令フォーマット(2)を示す図である。

【図13】同データ処理装置の第1の命令フォーマット(2)の命令のリストの一部を示す図である。

【図14】同データ処理装置において、第1のレジスタファイルの構成を示すブロック図である。

【図15】同データ処理装置の第1の命令フォーマットの命令の実行時のレジスタファイルのレジスタ番号を示す図である。

【図16】同データ処理装置の第2の命令フォーマットを示す図である。



【図17】同データ処理装置の第2の命令フォーマットの命令のリストの一部を示す図である。

【図18】同データ処理装置のレジスタファイルの構成を示すブロック図である。

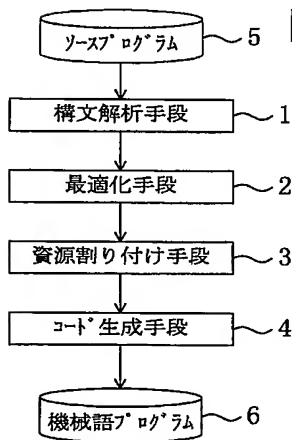
【図19】同データ処理装置において、第1の命令フォーマットの命令実行時のレジスタファイルのレジスタ番号を示す図である。

【図20】同データ処理装置において、第2の命令フォーマットの命令の実行時のレジスタファイルのレジスタ番号を示す図である。

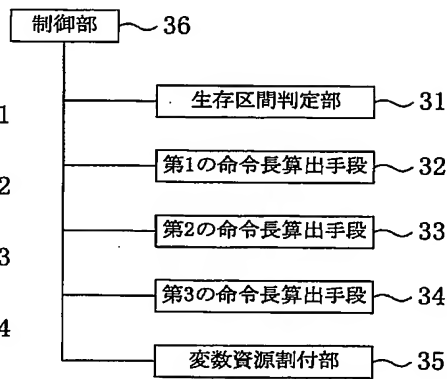
【符号の説明】

- 1 構文解析手段
- 2 最適化手段
- 3 資源割り付け手段
- 4 コード生成手段
- 31 生存区間判定部
- 32 第1の命令長算出手段
- 33 第2の命令長算出手段
- 34 第3の命令長算出手段
- 35 変数資源割り付け部
- 36 制御部

【図1】

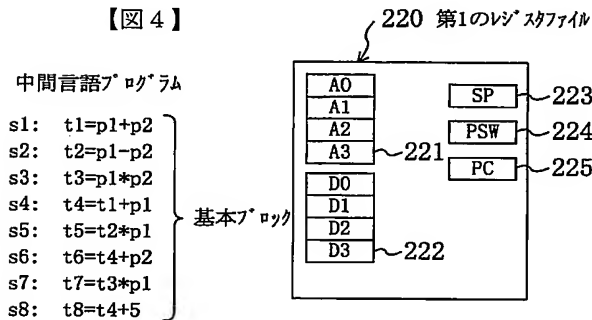


【図2】

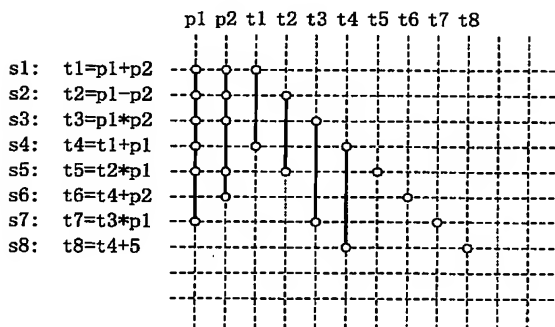


【図14】

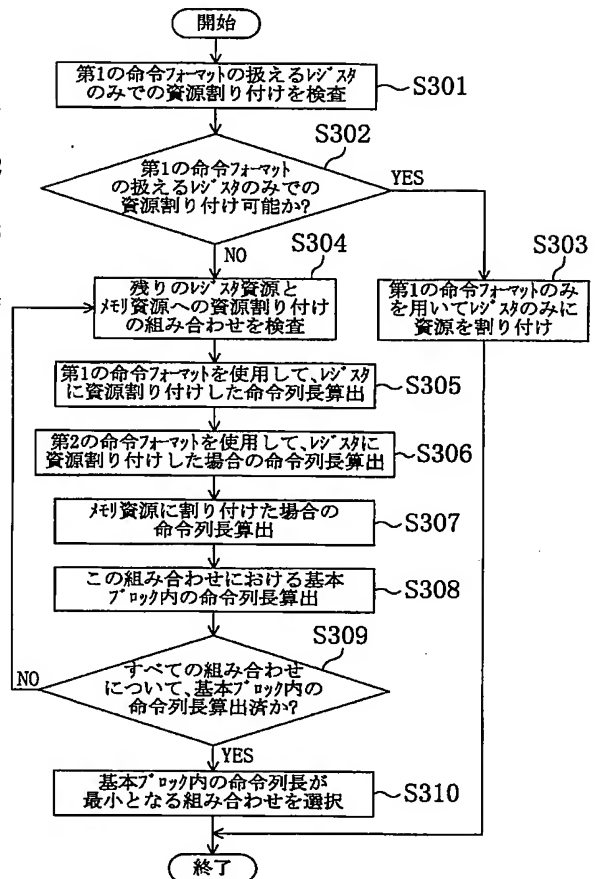
【図4】



【図5】



【図3】



【図15】

レジスタ名	命令コード上でのビット割り付け	物理的なレジスタ番号	物理的なレジスタ名
A0	00	00	アドレッシングレジスタ
A1	01	01	アドレッシングレジスタ
A2	02	02	アドレッシングレジスタ
A3	03	03	アドレッシングレジスタ
D0	00	00	データレジスタ
D1	01	01	データレジスタ
D2	02	02	データレジスタ
D3	03	03	データレジスタ

【図 6】

各変数の資源割り付け

p1	E0
p2	E1
t1	E2
t2	E3
t3	E4
t4	E5
t5	E6
t6	E7
t7	D2
t8	D3

中間言語プログラム

```

s1: t1=p1+p2
s2: t2=p1-p2
s3: t3=p1*p2
s4: t4=t1+p1
s5: t5=t2*p1
s6: t6=t4+p2
s7: t7=t3*p1
s8: t8=t4+5

```

機械語命令プログラム

```

s1: mov    E0, E2    3
    add    E1, E2    3

s2: mov    E0, E3    3
    sub    E1, E3    3

s3: mov    E0, E4    3
    mul    E1, E4    3

s4: mov    E0, E5    3
    add    E2, E5    3

s5: mov    E0, E6    3
    mul    E3, E6    3

s6: mov    E1, E7    3
    add    E5, E7    3

s7: mov    E0, D2    2
    mul    E4, D2    3

s8: mov    E3, D3    2
    add    5, D3     3

```

46byte

【図 7】

各変数の資源割り付け

p1	D2
p2	D3
t1	E0
t2	E1
t3	E2
t4	E3
t5	E4
t6	E5
t7	E6
t8	E7

中間言語プログラム

```

s1: t1=p1+p2
s2: t2=p1-p2
s3: t3=p1*p2
s4: t4=t1+p1
s5: t5=t2*p1
s6: t6=t4+p2
s7: t7=t3*p1
s8: t8=t4+5

```

機械語命令プログラム

```

s1: mov    D2, E0    2
    add    D3, E0    3

s2: mov    D2, E1    2
    sub    D3, E1    3

s3: mov    D2, E2    2
    mul    D3, E2    3

s4: mov    D2, E3    2
    add    E0, D3    3

s5: mov    D2, E4    2
    mul    E1, E4    3

s6: mov    D3, E5    2
    add    E3, E5    3

s7: mov    D2, E6    2
    mul    E2, E6    3

s8: mov    E3, E7    3
    add    5, E7     4

```

42byte

【図 8】

各変数の資源割り付け

p1	D2
p2	D3
t1	E0
t2	E1
t3	E2
t4	E3
t5	E4
t6	E5
t7	E6
t8	E7

中間言語プログラム

```

s1: t1=p1+p2
s2: t2=p1-p2
s3: t3=p1*p2
s4: t4=t1+p1
s5: t5=t2*p1
s6: t6=t4+p2
s7: t7=t3*p1
s8: t8=t4+5

```

機械語命令プログラム

```

s1: mov    D2, D0    1
    add    D3, D0    1
    mov    D0, E0    2

s2: mov    D2, D0    1
    sub    D3, D0    2
    mov    D0, E1    2

s3: mov    D2, D0    1
    mul    D3, D0    2
    mov    D0, E2    2

s4: mov    D2, D0    1
    add    E0, D0    3
    mov    D0, E3    2

s5: mov    D2, D0    1
    mul    E1, D0    3
    mov    D0, E4    2

s6: mov    D3, D0    1
    add    E3, D0    3
    mov    D0, E5    2

s7: mov    D2, D0    1
    mul    E2, D0    3
    mov    D0, E6    2

s8: mov    E3, D0    2
    add    5, D0     2
    mov    D0, E7    2

```

44byte

【図 9】

各変数の資源割り付け

p1	D2
p2	D3
t1	E0
t2	E1
t3	E2
t4	E3
t5	E4
t6	E5
t7	E6
t8	E7

中間言語プログラム

```

s1: t1=p1+p2
s2: t2=p1-p2
s3: t3=p1*p2
s4: t4=t1+p1
s5: t5=t2*p1
s6: t6=t4+p2
s7: t7=t3*p1
s8: t8=t4+5

```

機械語命令プログラム

```

s1: mov    D2, D0    1
    add    D3, D0    1
    mov    D0, E0    2

s2: mov    D2, E1    2
    sub    D3, E1    3

s3: mov    D2, E2    2
    mul    D3, E2    3

s4: mov    D2, E3    2
    add    E0, E3    3

s5: mov    D2, E4    2
    mul    E1, E4    3

s6: mov    D3, E5    2
    add    E3, E5    3

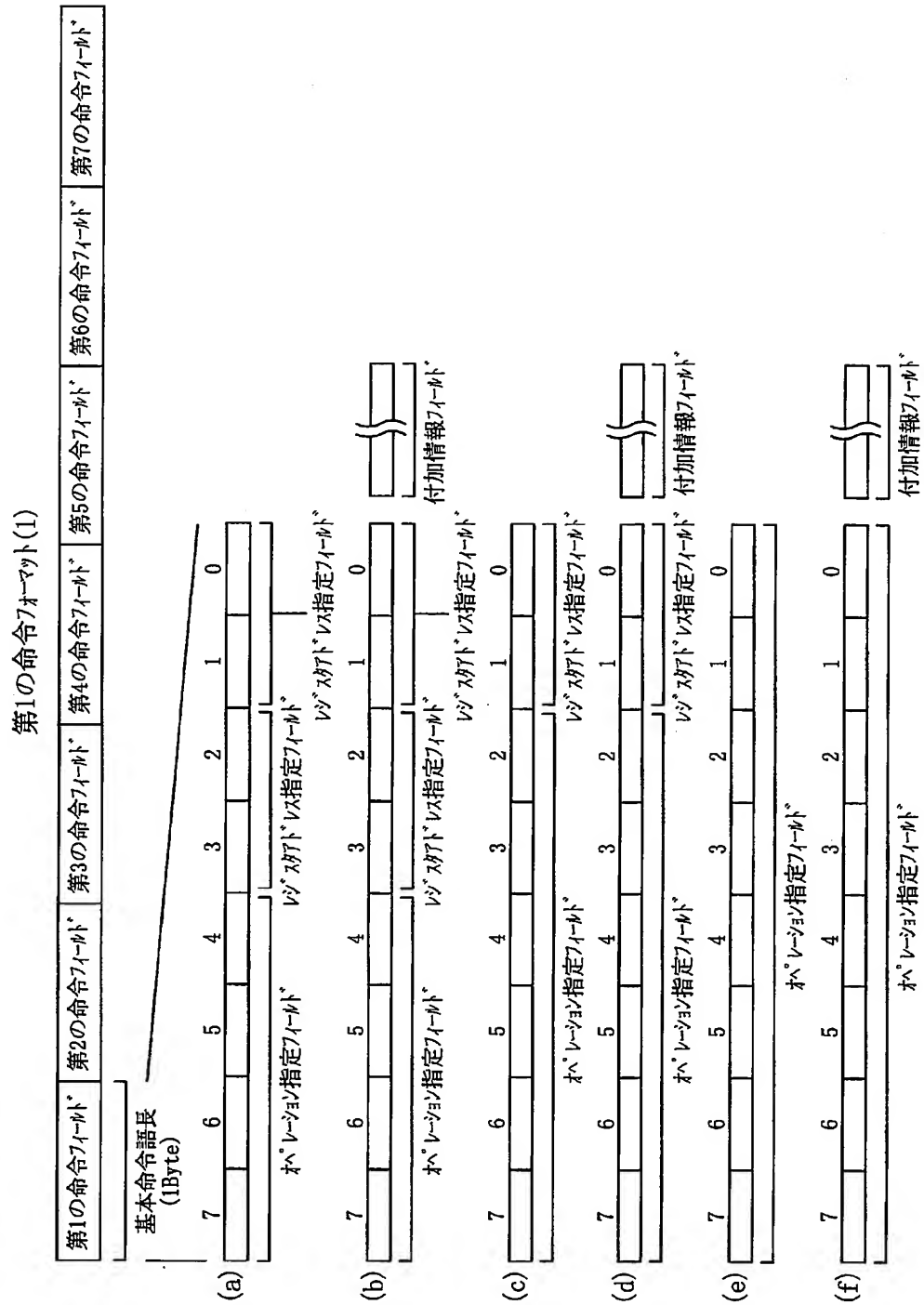
s7: mov    D2, E6    2
    mul    E2, E6    3

s8: mov    E3, D0    2
    add    5, D0     2
    mov    D0, E7    2

```

40byte

【図 10】



【図11】

## 第1の命令フォーマット(1)-(a)

ADD Dm, Dn : 加算  
 CMP Dm, Dn : 比較演算  
 CMP Am, An : 比較演算  
 MOV (Am), Dn : メモリ→レジスタ間転送(ロード)  
 MOV Dm, (An) : レジスタ→メモリ間転送(ストア)  
 MOV Dm, Dn : レジスタ→レジスタ間転送  
 MOV Am, An : レジスタ→レジスタ間転送  
 ...

## 第1の命令フォーマット(1)-(b)

...

## 第1の命令フォーマット(1)-(c)

CLR Dm : データ0クリア  
 INC Dm : データ1インクリメント  
 INC Am : データ1インクリメント  
 EXTB Dm : 符号付きバイトデータ→ワードデータ拡張  
 EXTUB Dm : 符号無しバイトデータ→ワードデータ拡張  
 EXTH Dm : 符号付きハーフワードデータ→ワードデータ拡張  
 EXTHU Dm : 符号無しハーフワードデータ→ワードデータ拡張  
 MOV SP, An : SP→レジスタ間転送  
 ...

## 第1の命令フォーマット(1)-(d)

ADD imm8, An : 8ビット即値加算  
 ADD imm8, Dn : 8ビット即値加算  
 MOV imm16, An : 16ビット即値転送  
 MOV imm16, Dn : 16ビット即値転送  
 MOV (abs16), Dn : 16ビット絶対アドレス指定によるメモリ→レジスタ間転送(ロード)  
 MOVBU (abs16), Dn : 16ビット絶対アドレス指定によるメモリ→レジスタ間転送(ロード)  
 /バイトデータ符号拡張  
 MOVHU (abs16), Dn : 16ビット絶対アドレス指定によるメモリ→レジスタ間転送(ロード)  
 /ワードデータ符号拡張  
 MOV Dm, (abs16) : 16ビット絶対アドレス指定によるレジスタ→メモリ間転送(ストア)  
 MOVBU Dm, (abs16) : 16ビット絶対アドレス指定によるレジスタ→メモリ間転送(ストア)  
 /バイトデータ符号拡張  
 MOVHU Dm (abs16) : 16ビット絶対アドレス指定によるレジスタ→メモリ間転送(ストア)  
 /ワードデータ符号拡張  
 MOV (disp8, SP), An : SP(スタックポインタ)相対指定によるメモリ→レジスタ間転送(ロード)  
 MOV (disp8, SP), Dn : SP(スタックポインタ)相対指定によるメモリ→レジスタ間転送(ロード)  
 MOV Am, (disp8, SP) : SP(スタックポインタ)相対指定によるレジスタ→メモリ間転送(ストア)  
 MOV Dm, (disp8, SP) : SP(スタックポインタ)相対指定によるレジスタ→メモリ間転送(ストア)  
 ...

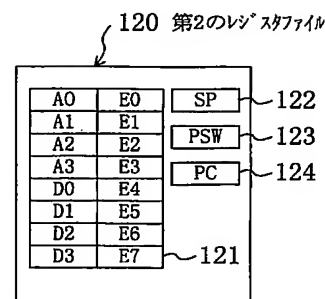
## 第1の命令フォーマット(1)-(e)

NOP : No Operation  
 LOOP #CC : ループ制御命令  
 ...

## 第1の命令フォーマット(1)-(f)

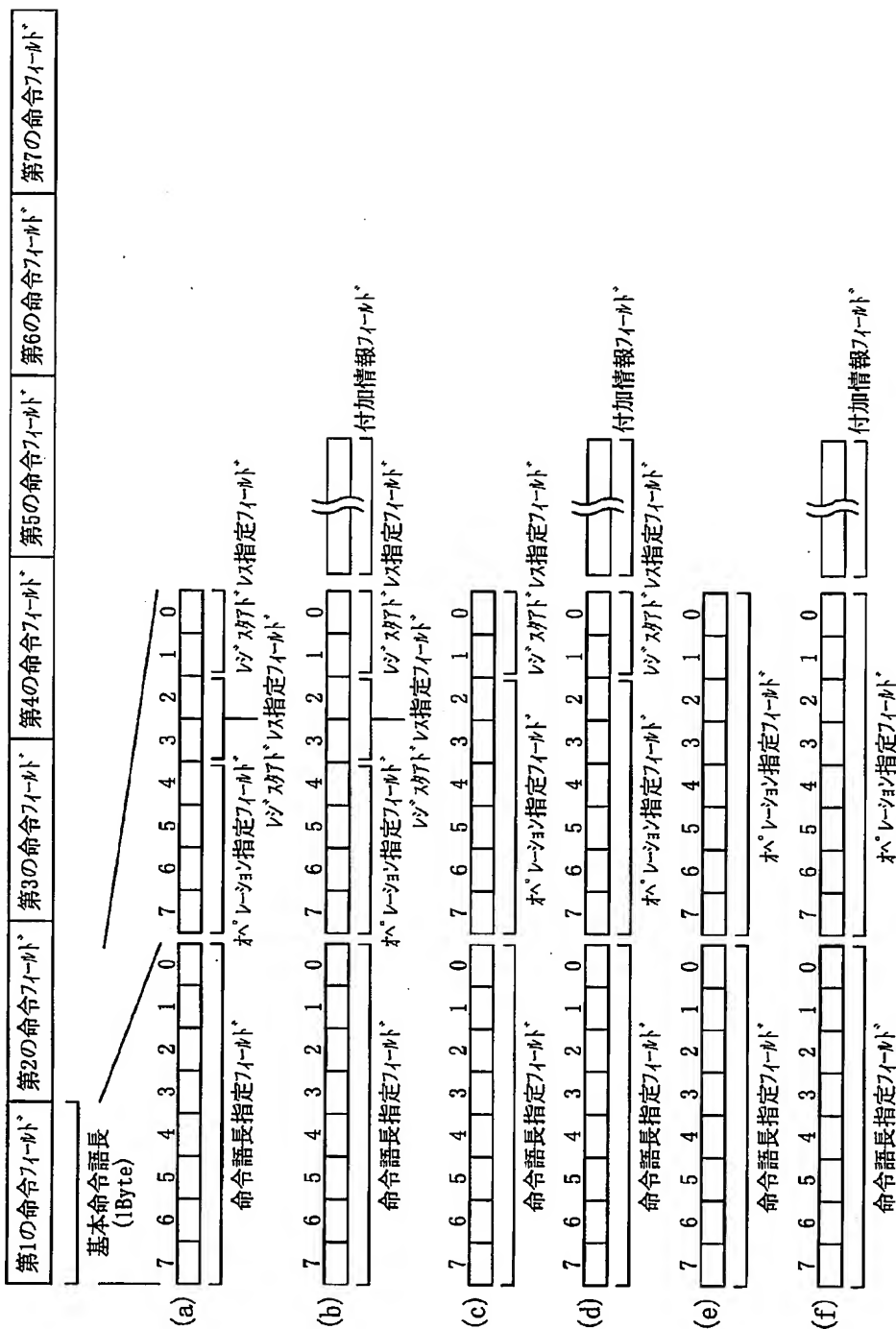
BR #CC(disp8, PC) : PC相対条件分岐  
 JMP (disp8, PC) : PC相対絶対分岐  
 ...

【図18】



【図12】

## 第1の命令フォーマット(2)



【図 1 3】

## 第1の命令フォーマット(2)-(a)

SUB Dm, Dn : 減算  
 MOV (Am), An : メモリ→レジスタ間転送(ロード)  
 MOV Am, (An) : レジスタ→メモリ間転送(ストア)  
 ...

## 第1の命令フォーマット(2)-(b)

MOV (Ai, Dn), Dn : インデックス付きレジスタ間接メモリ→レジスタ間転送(ロード)  
 ...

## 第1の命令フォーマット(2)-(c)

...

## 第1の命令フォーマット(2)-(d)

ADD imm16, An : 16ビット即値加算  
 ADD imm16, Dn : 16ビット即値加算  
 ...

## 第1の命令フォーマット(2)-(e)

RTI : 割り込み状態からの復帰  
 ...

## 第1の命令フォーマット(2)-(f)

...

【図 1 9】

レジスタ名	命令コード上での ビット割り付け	物理的な レジスタ番号	物理的な レジスタ名
A0	00	1000	汎用レジスタ
A1	01	1001	汎用レジスタ
A2	10	1010	汎用レジスタ
A3	11	1011	汎用レジスタ
D0	00	1100	汎用レジスタ
D1	01	1101	汎用レジスタ
D2	10	1110	汎用レジスタ
D3	11	1111	汎用レジスタ
E0			
E1			
E2			
E3			
E4			
E5			
E6			
E7			

【図 2 0】

レジスタ名	命令コード上での ビット割り付け	物理的な レジスタ番号	物理的な レジスタ名
A0	1000	1000	汎用レジスタ
A1	1001	1001	汎用レジスタ
A2	1010	1010	汎用レジスタ
A3	1011	1011	汎用レジスタ
D0	1100	1100	汎用レジスタ
D1	1101	1101	汎用レジスタ
D2	1110	1110	汎用レジスタ
D3	1111	1111	汎用レジスタ
E0	0000	0000	汎用レジスタ
E1	0001	0001	汎用レジスタ
E2	0010	0010	汎用レジスタ
E3	0011	0011	汎用レジスタ
E4	0100	0100	汎用レジスタ
E5	0101	0101	汎用レジスタ
E6	0110	0110	汎用レジスタ
E7	0111	0111	汎用レジスタ

## 第2の命令フォームット





【図17】

## 第2の命令フォーマット(a)

ADD Rm, Rn : 加算  
 SUB Rm, Rn : 減算  
 CMP Rm, Rn : 比較演算  
 MOV (Rm), Rn : メモリ→レジスタ間転送(ロード)  
 MOV Rm, (Rn) : レジスタ→メモリ間転送(ストア)  
 MOV Rm, Rn : レジスタ→レジスタ間転送  
 ...

## 第2の命令フォーマット(b)

ADD Rm, Rn, Rd : 加算  
 SUB Rm, Rn, Rd : 減算  
 MOV (Ri, Rm), Rn : インデックス付きレジスタ間接メモリ→レジスタ間転送(ロード)  
 ...

## 第2の命令フォーマット(c)

...

## 第2の命令フォーマット(d)

ADD imm 16, Rn : 16ビット即値加算  
 ADD imm 16, Rn : 16ビット即値加算  
 MOV (disp8, SP), Rn : SP(スタックポインタ)相対指定によるメモリ→レジスタ間転送(ロード)  
 MOV Rm, (disp8, SP) : SP(スタックポインタ)相対指定によるレジスタ→メモリ間転送(ストア)  
 ...